JACK: Java Auction Configuration Kit

Version 1.2

1 License

2 Downloading JACK

You can download the binary and source files for JACK by going to jack.cs.brown.edu and clicking on the "Download JACK" button. This zipped file contains the source and compiled files for the server, auctions, valueations, gui and example agents as well as the Dutch Flower Auction example. The configurable aspects of JACK (i.e. everything but the DFA) can be setup and run from the GUI. Instructions on how to run the GUI and edit the configuration files appear below. After these instructions there are further instructions on how to run a DFA game and interface with JACK.

3 Creating an Auction

3.1 Using JACK's GUI

JACK's GUI provides a way to create the required configuration files. To create a new game run the command ./JACK in the command line from the Jack folder. The gui will present a series of tabs that allow you to specify the auction schedule and parameters to construct your auction. The first tab provides the `Basic Setup` configuration information. Pick a name for your game. If your name is in use, you will be prompted to try again until you choose a unique name. You can also pick a minimum and maximum number of clients required for your game. Set the number of sequential and simultaneous auctions you would like to run and the total number of good types to be auctioned. You can also choose to use a local or public

IP address and can set the port to use for socket communication. 1300 is a good default port for most platforms.

After completing the `Basic Setup` tab, click "Next" to move on. This will save the setting you have entered. Currently, if you come back and edit this information, you will have to re-enter your specific auction, good and value information.

On the `Auction Schedule` tab you can select what type of sub-auction (1st Price, 2nd Price, Ascending or Descending) to run at a given slot. Auctions in columns will be run simultaneously and the set of auctions in a given column will be run before running the auction in the column to its right. (i.e. time runs left to right) You can select, assign and unselect and reassign auction as many times as you would like. You may also come back to this tab and make edits without them having any affect on future tab's information.

By clicking Next you will be taken to the `Goods Schedule` tab where you can set what good types will be auctioned in what slots and give names to good types. The matrix of good types on this tab corresponds directly to the auction matrix on the previous tab. Similarly to the `Auction Schedule` tab, you can edit this tab without affecting future tabs. However, you must click Next to add your input to the game state.

The next tab is the `Auction Parameters` tab. Here you can set values such as the reserve price, the starting price for a descending auction and the minimum increment value that an ascending auction requires new bids to surpass the current winning bid by to be acceptable.

Click "Next" to proceed to the valuation tabs. On the `Value Type` tab you can read about and select what valuation type you would like to use for your game. Then, after clicking" Next" you can set the different parameters to customize the prior over valuations that agents will receive on the `Value Parameters` tab.

After you have clicked Submit on the `Value Parameters` tab you have two choices.

1. You can save the game for later use by clicking "Save Game". A folder with the same unique name as the game will be created.

2. You can run the Server immediately. This will start the server and game play and feedback information will be printed in the shell you used to launch the GUI. The game will also be saved in a folder with the same unique name as the game.

The gui also allows you to load and run a game you previously saved. To do this click "Load Game". In the window that pops up, select the name of the game you want to run and click "Run Game" to start the server. Note, currently you cannot edit a saved game from the GUI. You can, however, edit these files by hand. A description of these files and their content appears below.

3.2 To Set Up or Configure the Files By Hand:

There are several configuration files:

1. In src/server:

   - Config_AuctionServer.txt
     This file allows you to pick/set:
     - Host_IP:Host's IP type (local, public)
     - Port_Number:pick a port number(1300 and above is generally safe on all platforms)
     - Min_Number_Clients:a minimum number of participating clients required to startcontinue an auction

- Max_Number_Clients:a maximum number of participating clients allowed in the auction
- Max_Wait_For_Clients:a maximum time that the server should wait for a client to join (in ms)
- Auction_Type:Super Auction type (SequentialAuction or SimultaneousAuction)
- Response_Time:60000
- Full_Response_Time:true
- Server_Log_File:Log_AuctionServer.txt
- Server_Results_File:Results_AuctionServer.txt
- Auction_Type:AuctionType
- Auction_Config_File:name of Auction_Type's configuration file
- Value_Type:Value type (Additive, Scheduling, Contracts)
- Value_Config_File:name of Value_Type's configuration file

2. In src/valuations:

- Config_ValuationAddative.txt
  This file allows you to pick/set 4 values for each item type to be sold:
  - lower bound for α, inclusive
  - upper bound for α, inclusive
  - lower bound for β, the exponent, inclusive
  - upper bound for β, the exponent, inclusive

  Values for alpha and beta will be drawn uniformly from the defined range for each agent. The agent's value will be calculated as $V = \alpha *(\text{number good type owned})^{\beta}$
  Example:

```
Good:Good1 Linear_Low_Bound:2 Linear_High_Bound:4 Exponent_Low_Bound:0.9 Exponent_Hig
Good:Good2 Linear_Low_Bound:5 Linear_High_Bound:10 Exponent_Low_Bound:0.9 Exponent_Hi
Good:Good3 Linear_Low_Bound:20 Linear_High_Bound:30 Exponent_Low_Bound:0.9 Exponent_H
```

- Config_ValuationContracts.txt
  This file allows you to pickset 4 values for each item type to be sold:
  - lower bound for number of the good needed to fulfill the contract
  - upper bound for number of the good needed to fulfill the contract
  - lower bound for value of the good in the contract
  - upper bound for value of the good in the contract

  Values for number of items and values will be drawn uniformly from the defined range for each agent. The agent's value will be calculated as
  $V =$ number of item*value of item $+ V$(rest of contract)
  Example:

```
Contracts_Per_Client:3
Good:Good1 Need_Low_Bound:1 Need_High_Bound:1 Value_Low_Bound:2 Value_High_Bound:4
Good:Good2 Need_Low_Bound:0 Need_High_Bound:1 Value_Low_Bound:5 Value_High_Bound:10
Good:Good3 Need_Low_Bound:0 Need_High_Bound:1 Value_Low_Bound:20 Value_High_Bound:30
```

- Config_ValuationScheduling.txt
  This file allows you to pickset 4 values for each item type to be sold:
  - lower bound for value of completing the job.
  - upper bound for value of completing the job.
  - lower bound for number of goods required to complete the job.
  - upper bound for number of goods required to complete the job.
  - lower bound for deadline for completing the job.
  - upper bound for deadline for completing the job.

  Values for number of goods required, deadlines and values will be drawn uniformly from the defined range. Values will be sorted to be monotonically decreasing. After its deadline the agent will receive 0 value. The agent's value will be calculated as
  $V = v_i$ where i is the number of the item where the sum of the items
  Example:

  ```
  Value_Low_Bound:0 Value_High_Bound:50 Number_Of_Goods:5 Need_Low_Bound:1 Need_High_Bound:5
  ```

3. In src/auctions:

- Config_Sequential_of_SimultaneousAuction.txt
  The first line specifies the type of valuation, the configuration file for the valuation and the number of contracts each agent should get if you are using contract valuations.

  Examples:
  Valuation:Contracts Config_File:Config_ValuationContracts.txt Contracts_Per_Client:5
  Valuation:Additive Config_File:Config_ValuationAdditive.txt

  Next, create a line for each Simultaneous auction you want run.
  - Auction_Type:SimultaneousAuction Config_File:[Specific Simultaneous Auction Config File].txt
- Config_SequentialAuction.txt
  Config_SimultaneousAuction.txt
  The first line specifies the type of valuation, the configuration file for the valuation and the number of contracts each agent should get if you are using contract valuations.

  Examples:
  Valuation:Contracts Config_File:Config_ValuationContracts.txt Contracts_Per_Client:5
  Valuation:Additive Config_File:Config_ValuationAdditive.txt

  Create a line for each auction you want run, in the order you want the auctions to be run.
  - Auction_Type:[TypeOfAuction] Config_File:[Specific Auction Config File].txt

- Config_FirstPriceAuction.txt
  Config_SecondPriceAuction.txt

  Here, set the Good (a string that identifies the object) and the Reserve price.

  - Good:good2 Reserve:3

- Config_AscendingPriceAuction.txt
  Config_DescendingPriceAuction.txt

  Here, set the Good (a string that identifies the object), the Reserve price, the Start_Price for descending and Min_Price_Increment for ascending.

  - Good:Good2 Reserve:3 Start_Price:100
  - Good:Good1 Reserve:3 Min_Price_Increment:1

  Note: You may have several variations of an auction type i.e. with different reserve prices or Goods. Indicate these with unique file names and use the Sequential/Simultaneous files to state what these files are and in what order the corresponding auctions should be run.

3.3 Compile and Run java files:

EITHER:
If you are using unix, run ./buildAll to recompile all project folders. This script will recognize if you are just compiling clients.

OR:
To run on unix:
For the Gui (from which you can run the server):
run ./runJACK from the command line
For just the server(note: you must have configuration files correctly saved):
run ./startServer from the command line
OR:
java -cp bin server.AuctionServer

4 To Run the Server From the Eclipse IDE

From Eclipse:

"File" → New → JavaProject
Project Name: your choice
"Finish"

File → Import → General → fileSystem → "Browse..." → Brown_Auction_Framework
Select the whole folder
"Import"

There will now be packages for each of the modules(auctions, clients, server). To run the server open and run the AuctionServer.java class.

Eclipse will compile the server. Note, one instance of Eclipse cannot run the clients and server.

5 To Run a Client

Whether you are running the server from eclipse or the command line you will be running clients from the command line.

There is a file named IP_and_Port.txt. Ensure that the IP address in this file is the same as the one that the server prints when it is started from the location where it will be running. Place this file at the same level as the src and bin directories.

To compile clients run:
EITHER
```
./buildAll
```
OR
```
javac -d bin -cp bin src/clients/*.java
```

To run the client run:
EITHER
```
./startAClient [clientClass]
```
OR
```
java -cp bin clients.[clientClass]
```

AN EXAMPLE:
```
./startAClient CmdLineClient
```
will create a client a human can interact with.